Automated run-time regression testing with Fuego

21 Aug 2019

Hirotaka MOTAI

MITSUBISHI ELECTRIC
Changes for the Better

Outline

- Who I am
- Overview
- Related Tools
  - Automated Test System / Fuego
  - Linux Test Project / LTP
- Issue
- Approach
- Conclusion and Future work

Who I am

- Hirotaka MOTAI
  - Software researcher for embedded systems of MITSUBISHI ELECTRIC Corp.

- We have collaborated with LF projects.
  - **LTSI**: Long Term Support Initiative
  - **AGL**: Automotive Grade Linux
  - **Fuego**: Automated Test System
    - specifically designed for testing Embedded Linux

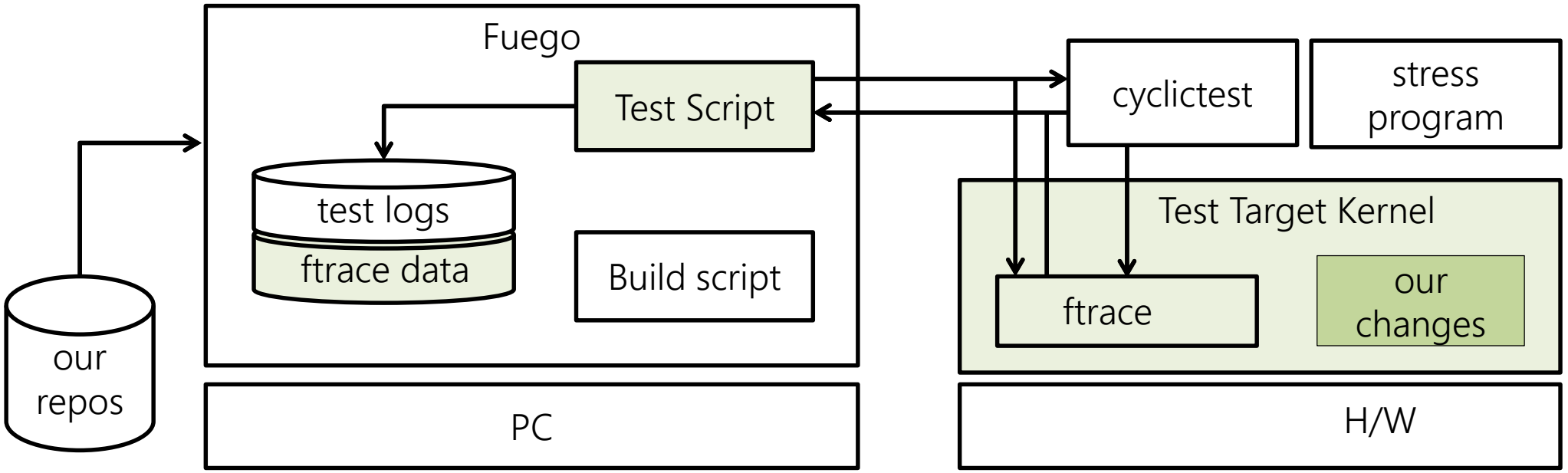
Overview

- Linux can be adapted to various embedded devices, even though they need a hard real-time response.

- We need tons of time to ensure adequate real-time performance.
  - Real-time applications need to satisfy timing constraints.
  - We have to avoid kernel changes which might cause long delays.

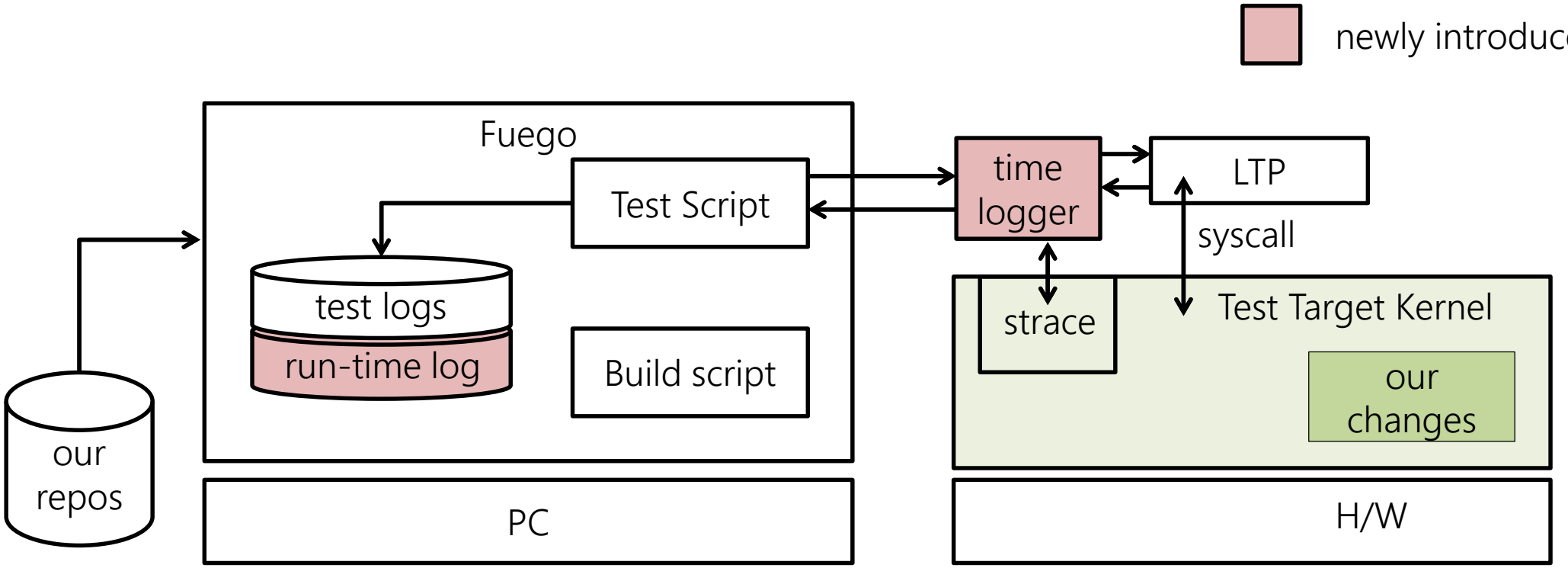
Overview

- Detect and Ready for analysis performance issue in Automated Testing Framework.
  - In our use case with “Fuego” (presented in ELCE2018)
  - measure the real-time performance, plus get tracing.
  - get clues to distinguish the problem whether it was caused by our changes or not.



Overview

- We have developed a part of Functional-test run-time logger to get clues to detect internal performance problems even if all of the function test are successful.



MITSUBISHI ELECTRIC
Changes for the Better

RELATED TOOLS

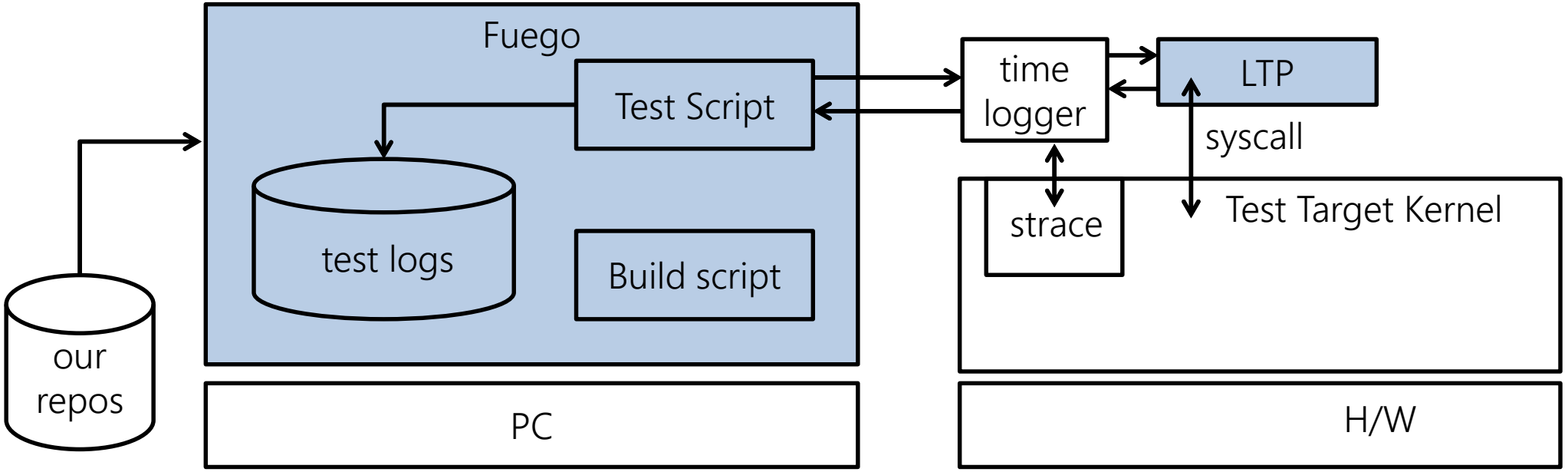
Fuego

LTP

Related Tools

- Fuego:
  - an automated test system specifically designed for embedded Linux testing
  - http://fuegotest.org/

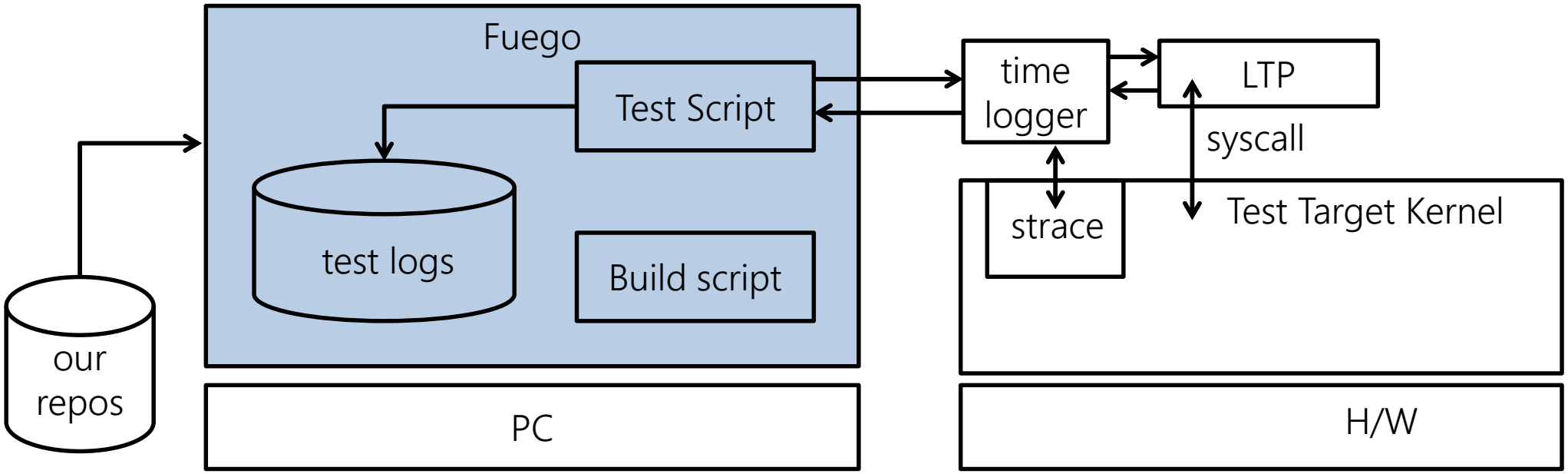
- LTP: Linux Test Project
  - regression and conformance tests designed to confirm the behavior of the Linux kernel and glibc
  - http://linux-test-project.github.io/



Fuego

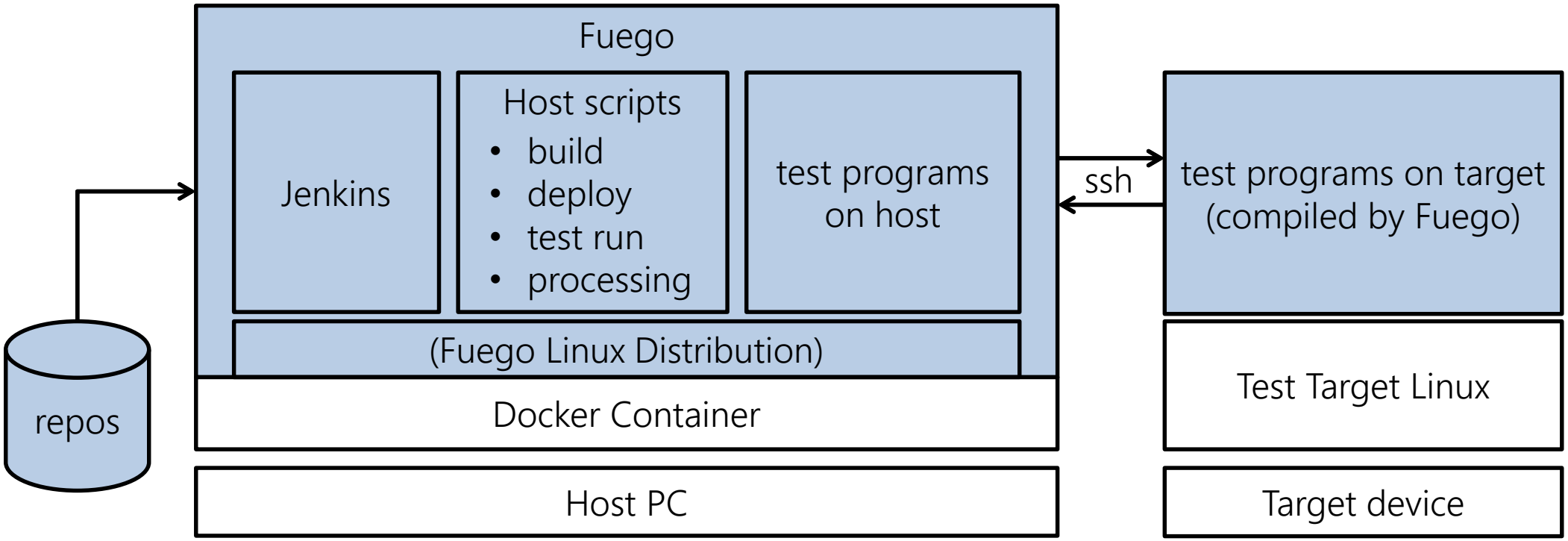
JTA: Jenkins Test Automation

- Fuego is an automated test system
  - created by LTSI project, based on Jenkins.
  - OSS: anyone can use and contribute!
  - AGL-JTA: AGL chose Fuego as standard test environment.



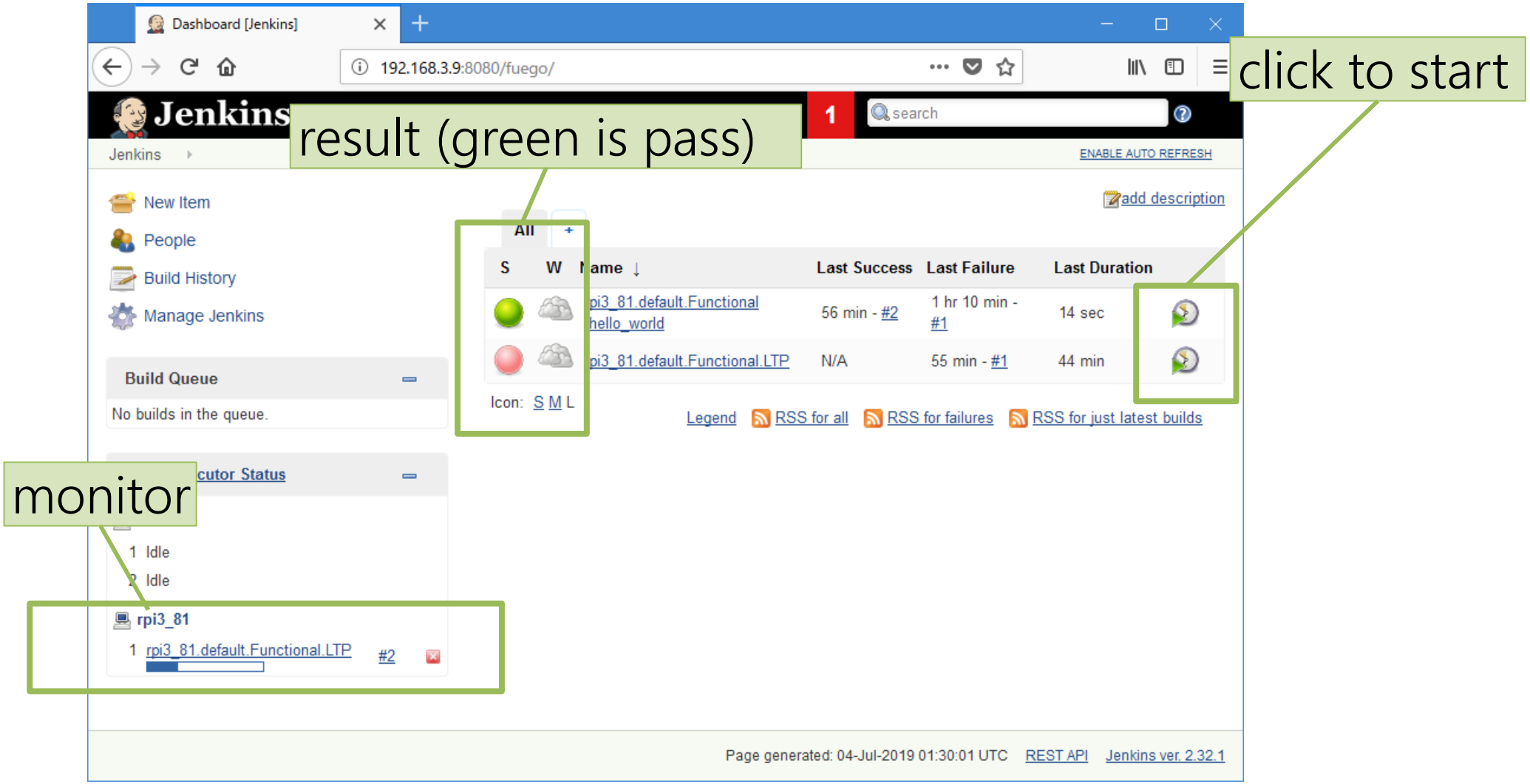
Fuego

- Fuego = "test distribution + Jenkins + host scripts + pre-packaged tests" on container
- Features: test code build, deploy, run, results report.
  - simple board setup, running tests in batches …



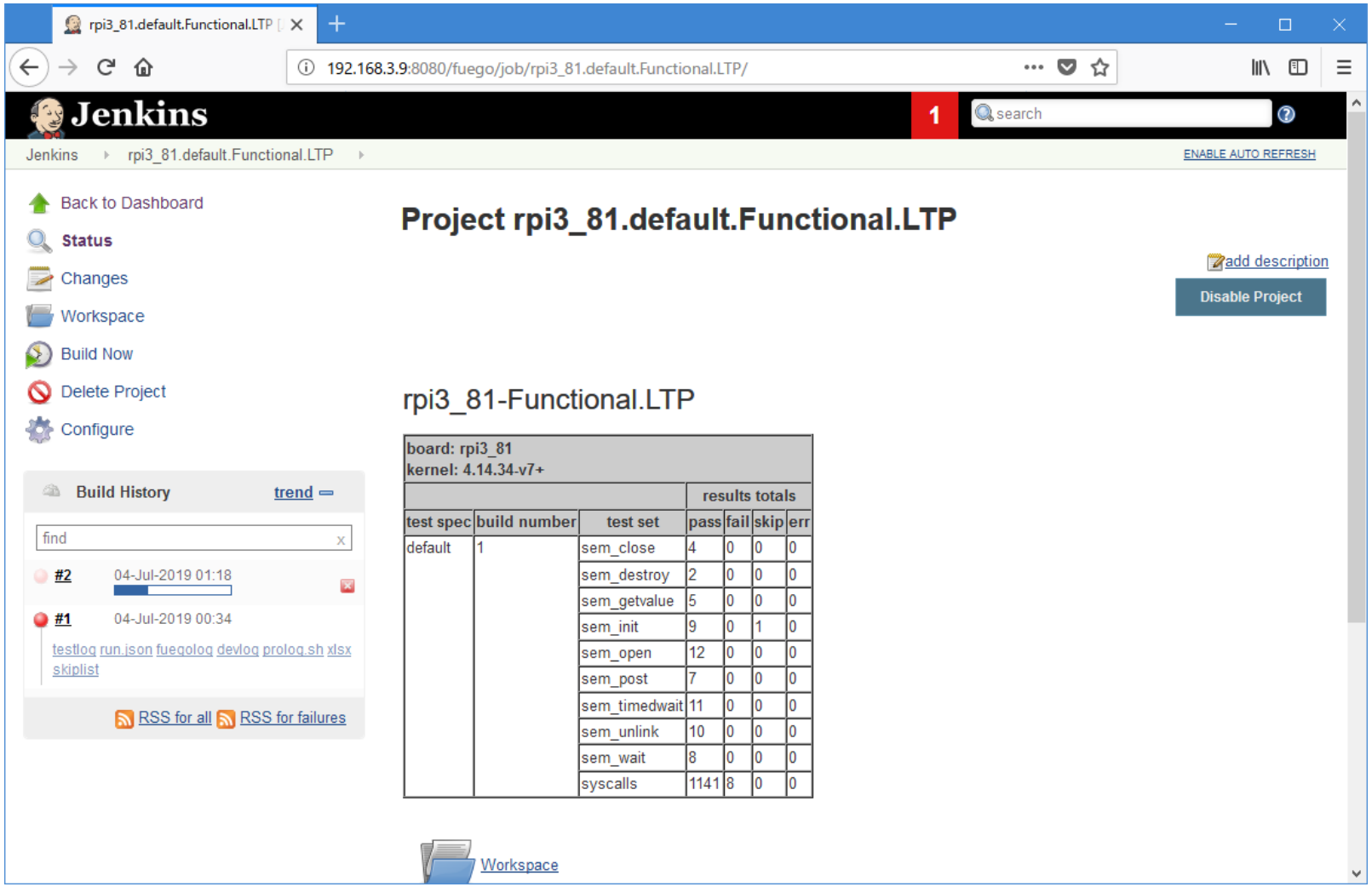
Fuego

- You can click to start manually and monitor tests on Jenkins.



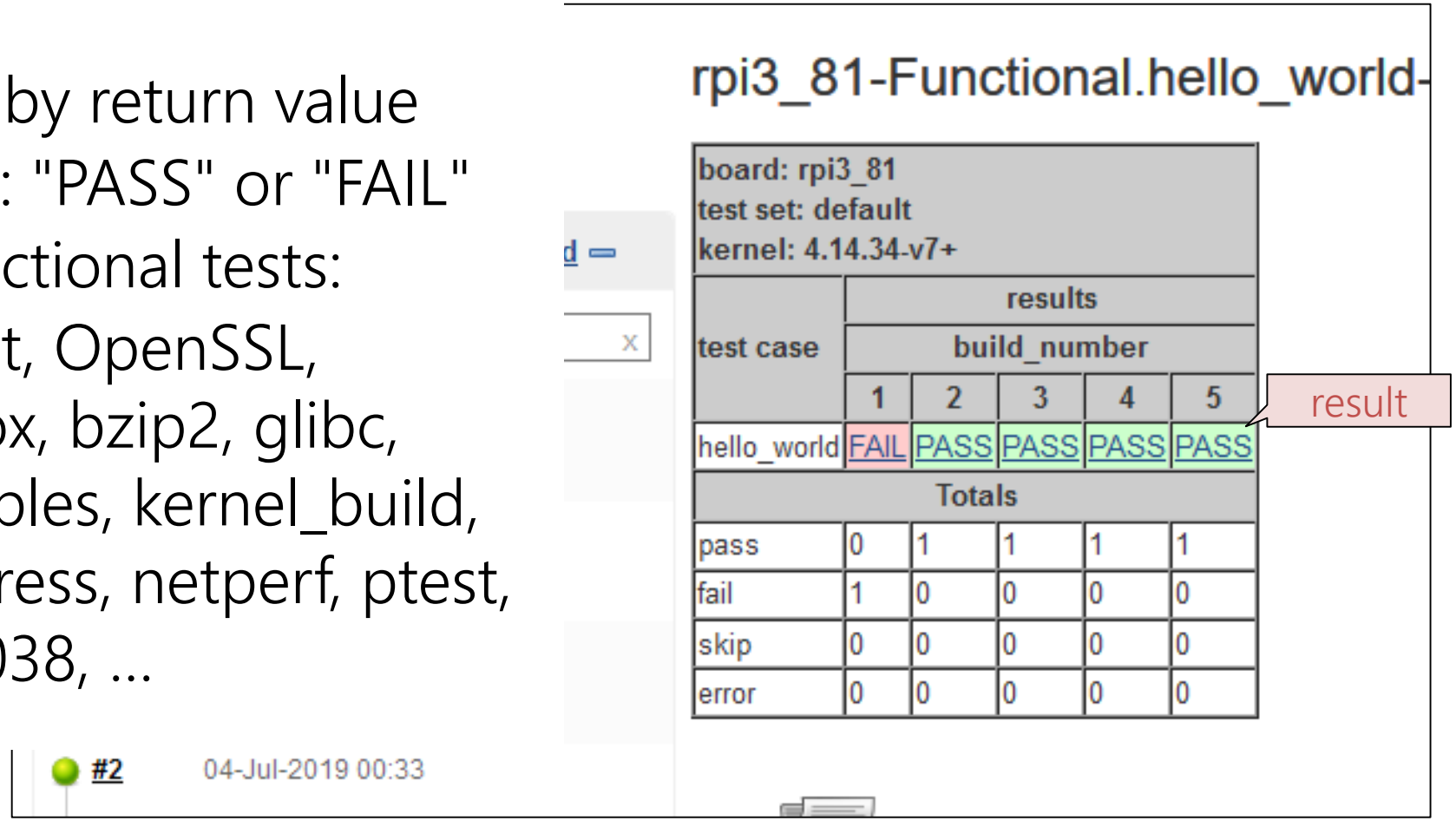
Fuego

- You can also check test results on Jenkins.



Fuego

- Functional test
  - Test result: judged by return value
    - Historical results: "PASS" or "FAIL"
  - 102 testsuits as functional tests:
    - LTP, LTP_one_test, OpenSSL, aiostress, busybox, bzip2, glibc, hello_world, iptables, kernel_build, kselftest, linus_stress, netperf, ptest, stress, tar, year2038, ...



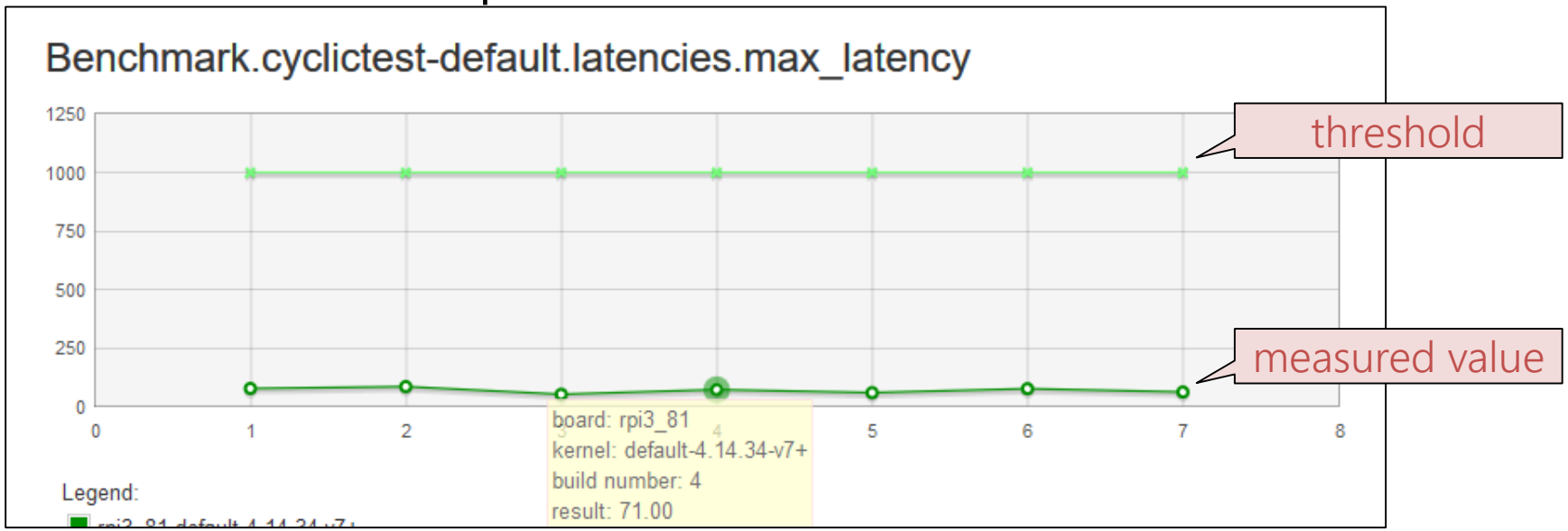
rpi3_81-Functional.hello_world-

board: rpi3_81
test set: default
kernel: 4.14.34-v7+

test case	results				
	build_number				
	1	2	3	4	5
hello_world	FAIL	PASS	PASS	PASS	PASS
Totals					
pass	0	1	1	1	1
fail	1	0	0	0	0
skip	0	0	0	0	0
error	0	0	0	0	0

Fuego

- Benchmark test
  - Test result: "PASS" if measured value < threshold
    - Historical results: measured value
  - 42 testsuits as benchmark tests:
    - Dhrystone, IOzone, Interbench, Whetstone, bonnie, cyclictest, dbench4, deadlinetest, hackbench, iperf, lmbench2, nbench_byte, netperf, svsematest, x11perf …

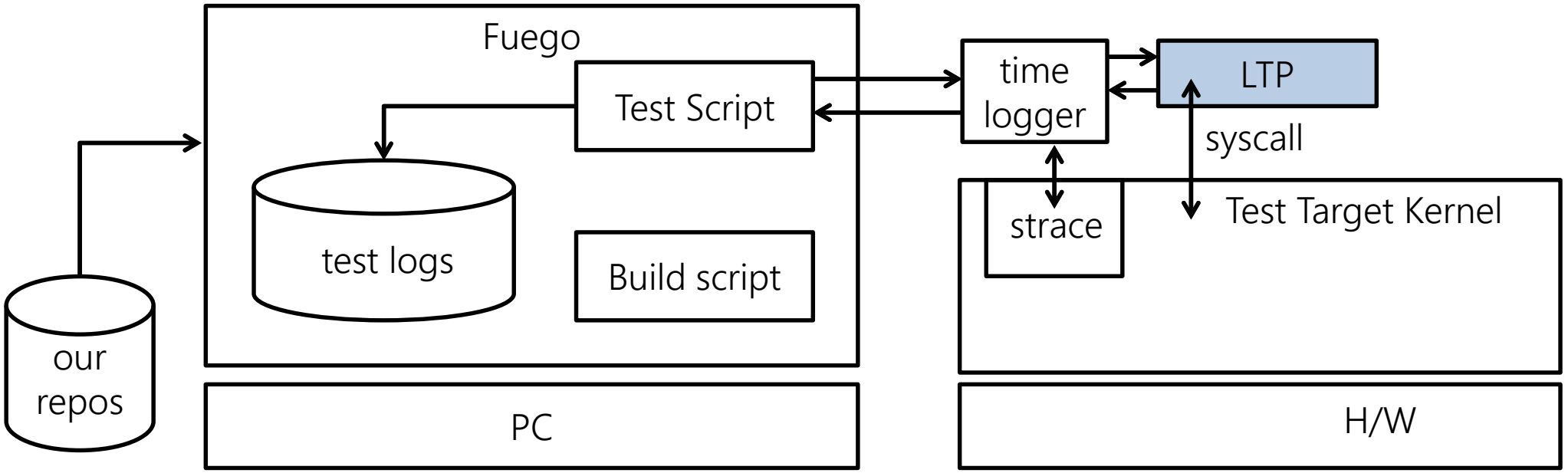


LTP: Linux Test Project

- A huge collection of tests for Linux
  - systemcalls, semaphore, POSIX, …
- Difficult to understand test results
  - Tester has to know what to ignore, and why
    - depend on system or kernel configurations.
  - In a regression test, tester check the gaps between previous and current results.

LTP on Fuego

- Fuego has 2 categories related to LTP
  - Functional.LTP
    - 14 test scenarios with using LTP test suit
  - Functional.LTP_one_test
    - only one LTP test that you can define with using LTP test suit



MITSUBISHI ELECTRIC
Changes for the Better

ISSUE & APPROACH

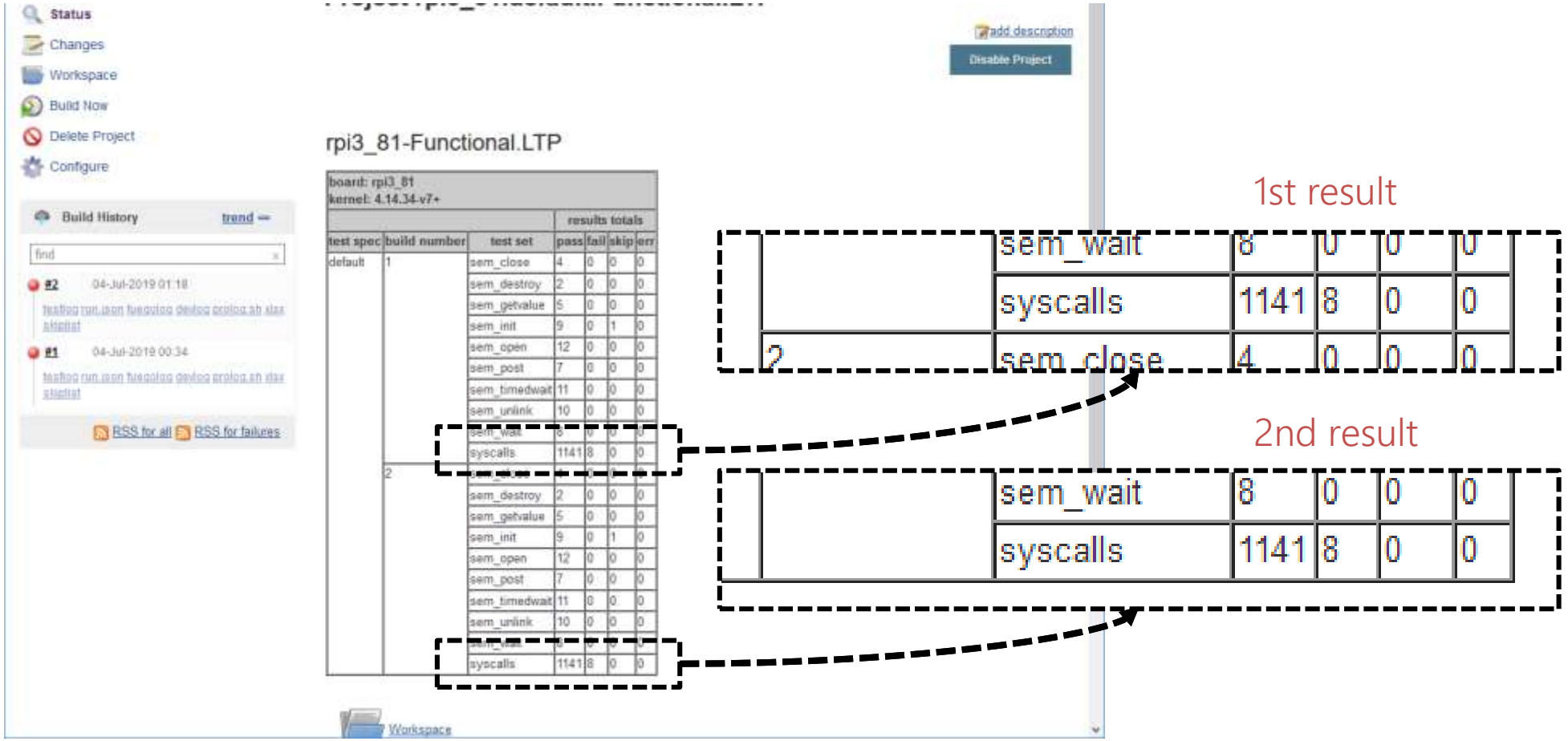
Issue

- Focus on syscall interface for checking regression
  - Influence performance of real-time process directly

- LTP can test syscall interfaces.
  - LTP on Fuego is helpful for checking compatibility

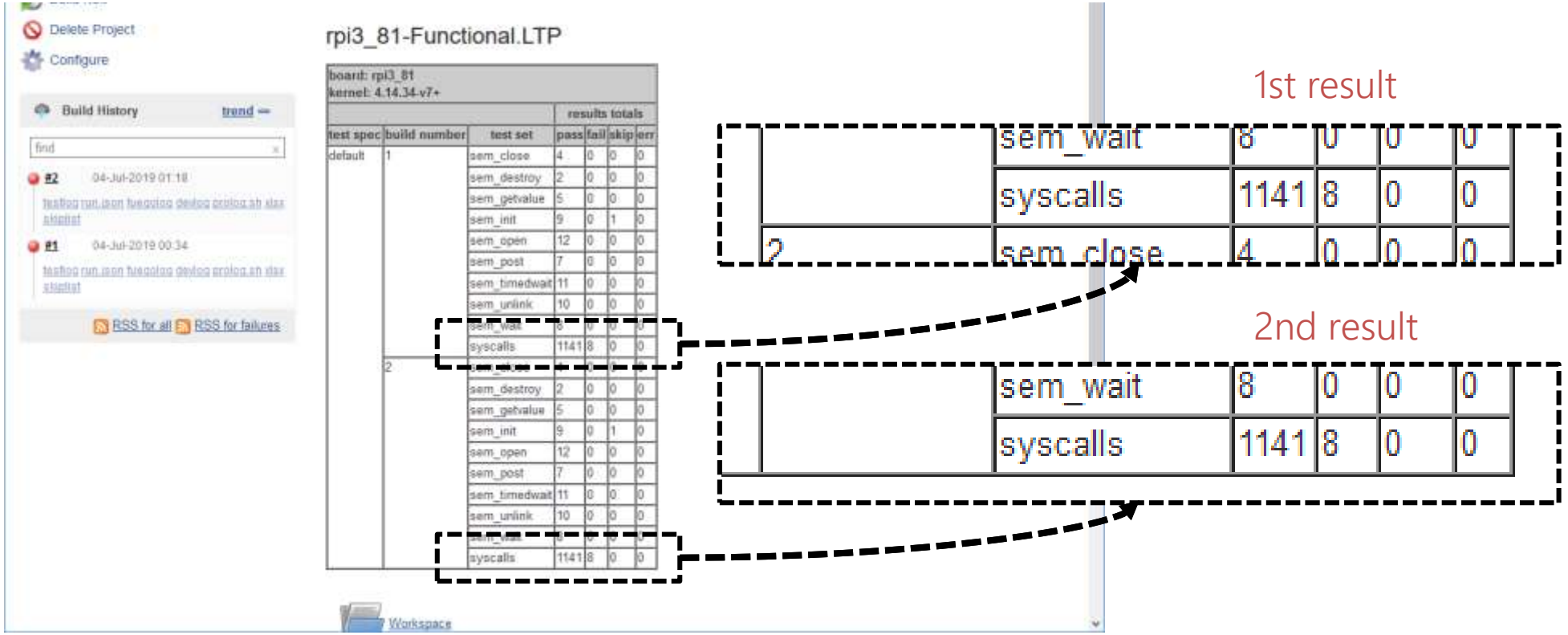
Issue

- Results for syscall tests look same…
- In term of regression check, looks good…..?



Issue

- It is important to make the difference clear.
  - What syscalls were "pass"ed? Is the results same?
  - Were new results "execution time of each syscall" as same as previous one?



Alternative way

- Using LTP_one_test in Fuego with some modifications
  - list our important syscall in spec.json

- add jobs

```
# ftc add-jobs -b rpi3_81 ¥
    -t Functional.LTP_one_test  ¥
    -s syscalls-shmat01
```

- build jobs

```
# ftc build-jobs ¥
    rpi3_81.syscalls-*.Functional.LTP_one_test
```

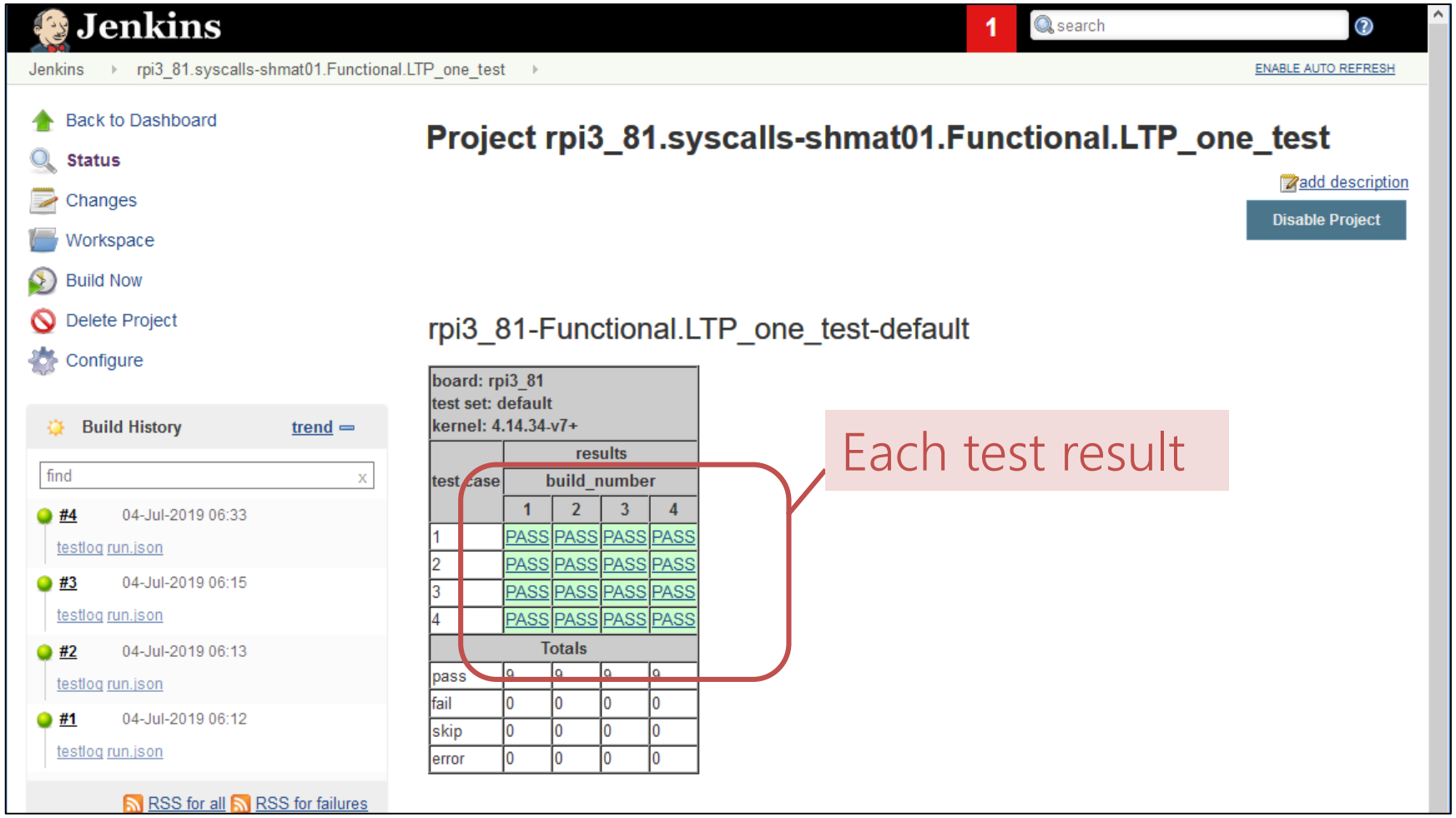
- Sample: shmat(), shmdt()

```
 {
    "testName": "Functional.LTP_one_test",
    "specs": {
      "default": {
        "TEST":"brk01"
      },
<SNIP>
      },
+     "syscalls-shmat01": { "TEST":"shmat01" },
+     "syscalls-shmat02": { "TEST":"shmat02" },
+     "syscalls-shmdt01": { "TEST":"shmdt01" },
+     "syscalls-shmdt02": { "TEST":"shmdt02" },
      "syscalls-mlock03": {
        "TEST":"mlock03",
        "scenario":"syscalls"
      }
    }
  }
```

ftc: "fuego test control" tool. a command line tool used to perform various functions in Fuego.

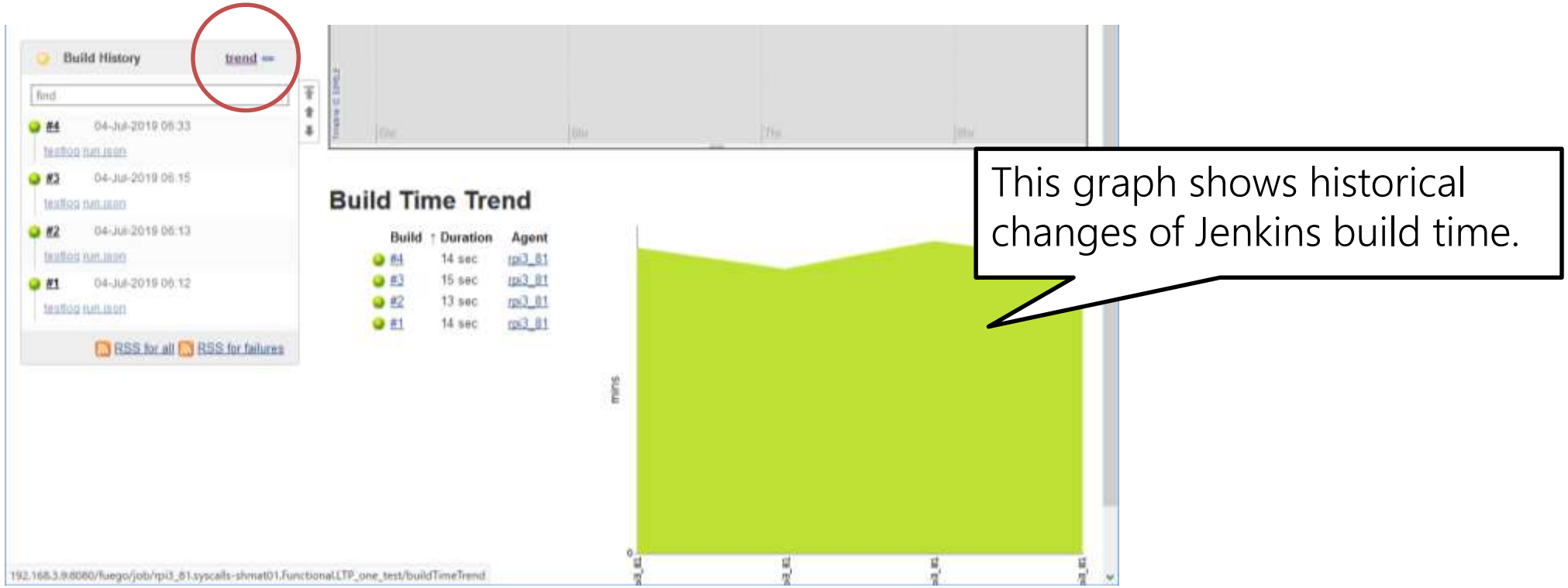
Alternative way

- Gap of test result of each syscall become clear.



Alternative way

- Gap of test result of each syscall become clear.
- However each execution time has not been clear yet.
  - the figure below shows Build Time Trend, not the execution time of syscall.



How to check the syscall time

- Do in a simple way.
  - Fuego provides a script running on the target, in fuego_test.sh.
  - measure the execution time of the test process as below.

```
 function test_run {
    local bdir="$BOARD_TESTDIR/fuego.$TESTDIR"
    local scenario=$FUNCTIONAL_LTP_ONE_TEST_SCENARIO

    if [ -z "$scenario" ] ; then
-       report "cd $bdir; ./$one_test $FUNCTIONAL_LTP_ONE_TEST_ARGS
+       report "cd $bdir; ./runtime-logger.sh ./$one_test $FUNCTIONAL_LTP_ONE_TEST_ARGS
    else
        report "cd $bdir; ./runltp -f $scenario -s $one_test"
    fi
 }
```

How to check the syscall time

- Do in a simple way.
  - Fuego provides a script running on the target, in fuego_test.sh.
  - measure the execution time of the test process as below.

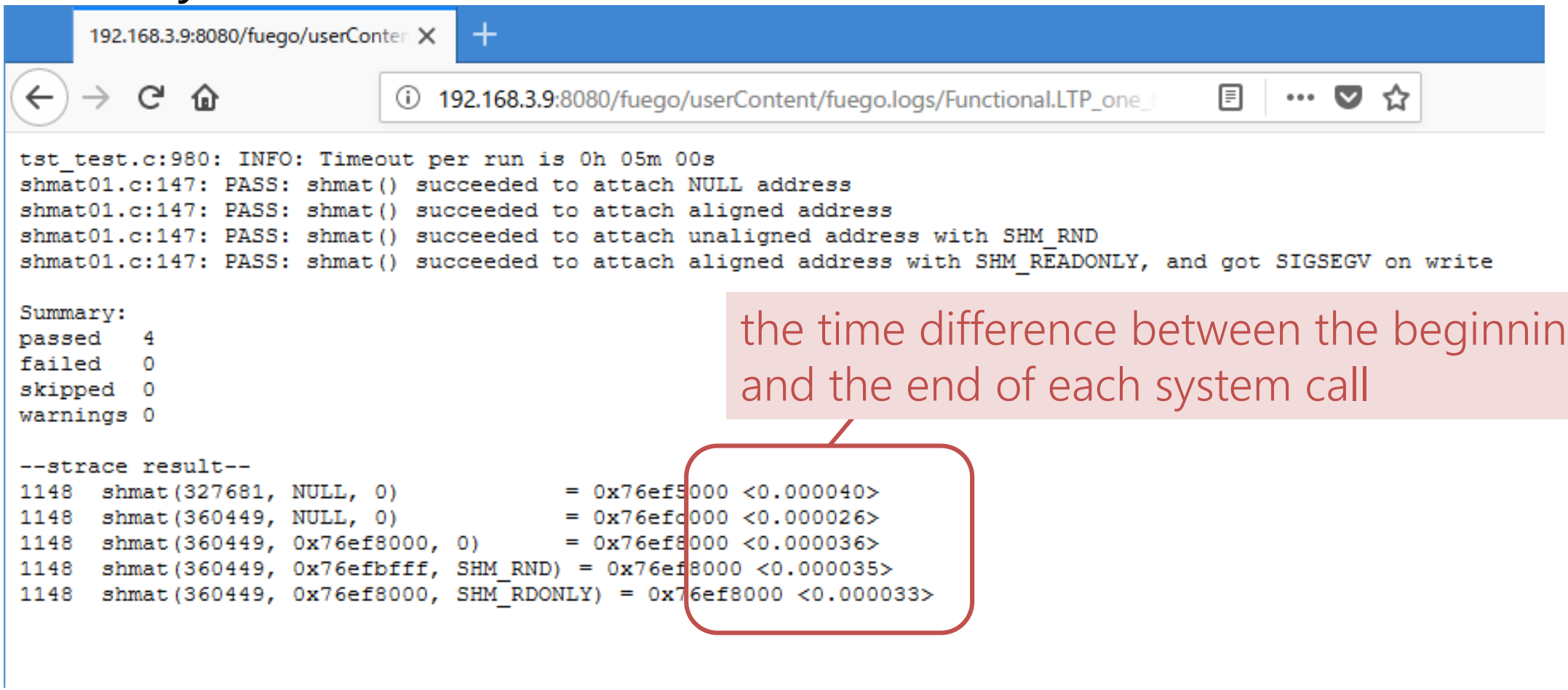
```
## runtime-logger.sh
SYSCALL=$(echo $1 | sed -e "s:^./::" -e "s:[0-9].*::")
OUTPUT=strace_${1##*/}.log
strace -f -T -e $SYSCALL -o $OUTPUT $* ; RETVAL=$?
echo -e "¥n--strace result--";
grep $SYSCALL $OUTPUT
exit $RETVAL
```

```
 function test_run {
    local bdir="$BOARD_TES
    local scenario=$FUNCTI

    if [ -z "$scenario" ] ; then
-       report "cd $bdir; ./$one_test $FUNCTIONAL_LTP_ONE_TEST_ARGS
+       report "cd $bdir; ./runtime-logger.sh ./$one_test $FUNCTIONAL_LTP_ONE_TEST_ARGS
    else
        report "cd $bdir; ./runltp -f $scenario -s $one_test"
    fi
 }
```

How to check the syscall time

- The execution time of the test process is saved with 1usec accuracy



Evaluation

- Confirmation
  - Inject 1sec waiting patch to "shmat()" interface in kernel.
  - Test and check whether the result include >1sec delay.

```
 long do_shmat(int shmid, char __user *shmaddr, int shmflg,
               ulong *raddr, unsigned long shmlba)
 {
        struct shmid_kernel *shp;
<<snip>>
        unsigned long populate = 0;

+       ssleep(1);
+
        err = -EINVAL;
        if (shmid < 0)
                goto out;
```

Evaluation

- The different time can be detected in the result

Injected Kernel

192.168.3.9:8080/fuego/userContent/fuego.logs/Functional.LTP_one_t

```
tst_test.c:980: INFO: Timeout per run is 0h 05m 00s
shmat01.c:147: PASS: shma
shmat01.c:147: PASS: shma
shmat01.c:147: PASS: shmat() succeeded to attach unaligned address with SHM_RND
shmat01.c:147: PASS: shmat() succeeded to attach aligned address with SHM_READONLY, and got SIGSEGV on write

Summary:
passed   4
failed   0
skipped  0
warnings 0

--strace result--
1074  shmat(327682, NULL, 0)             = 0x76fd8000 <1.012371>
1074  shmat(360450, NULL, 0)             = 0x76fdf000 <1.039117>
1074  shmat(360450, 0x76fd8000, 0)       = 0x76fd8000 <1.020016>
1074  shmat(360450, 0x76fdbfff, SHM_RND) = 0x76fd8000 <1.037206>
1074  shmat(360450, 0x76fd8000, SHM_RDONLY) = 0x76fd8000 <1.037140>
```

Each result was "PASS" as same as in default kernel.

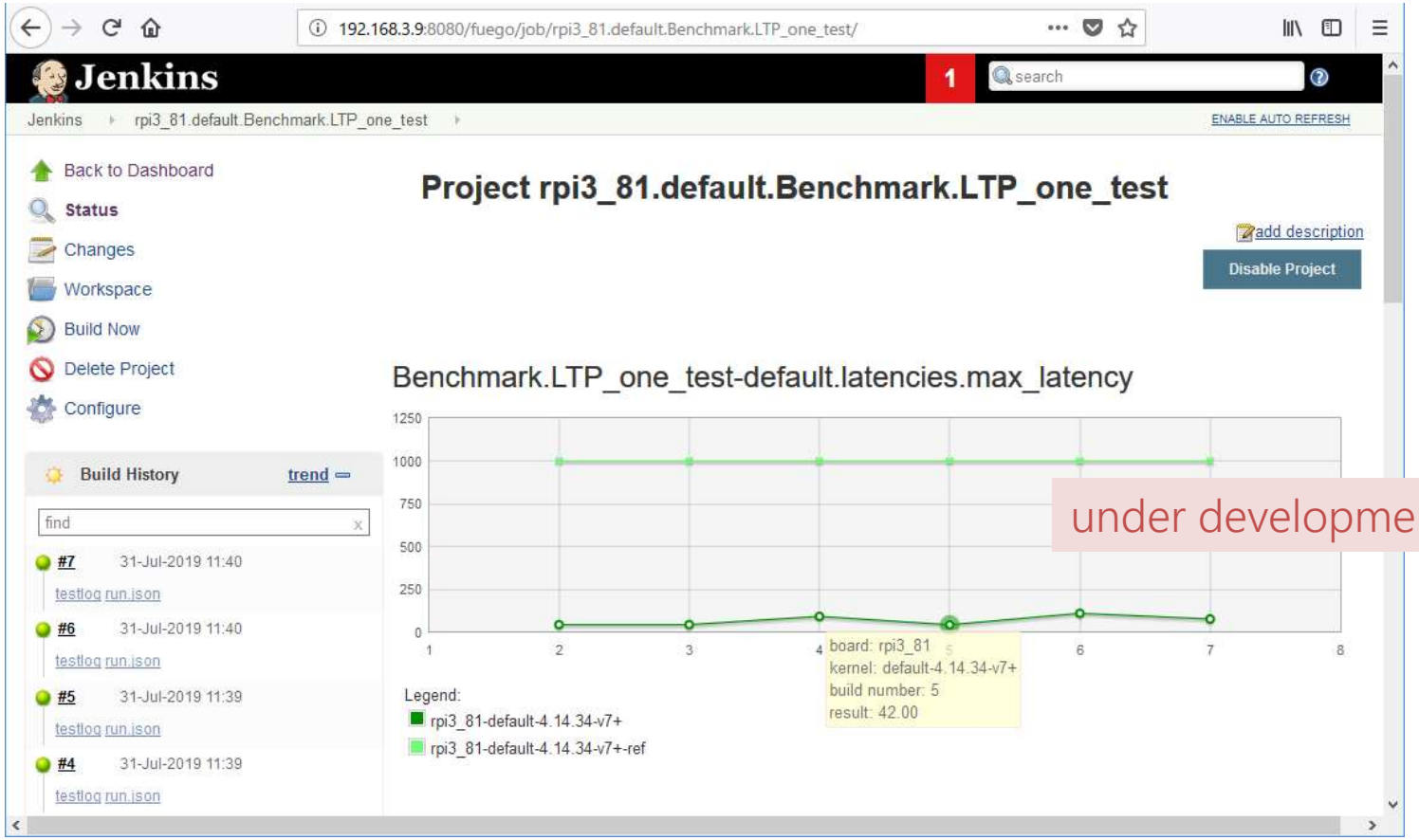
The time differences compared with the result in defalt kernel were roughly "1 second" each system call.

Conclusion and Future work

- Summary
  - Real-time applications need to satisfy timing constraints.
    - In term of regression, syscall time in new Linux will be shorter or as same as old one.
  - Fuego is useful to us for not only functional checking but also measuring to syscalls.

Conclusion and Future work

- Future works
  - Visualization: line graph of measurement time
    - Discussed this idea at Fuego Jamboree #3 (20 July 2019)
    - Current status:
      Developing it
      as Benchmark test,
      not Functional test.



under development

MITSUBISHI ELECTRIC
Changes for the Better

THANK YOU!

Any Questions?

MITSUBISHI ELECTRIC
Changes for the Better

APPENDIX

Resources

- FUEGO
  - http://fuegotest.org/
- LTP: Linux Test Project
  - http://linux-test-project.github.io/
- strace
  - https://strace.io/
- LTSI Project
  - https://ltsi.linuxfoundation.org/
- AGL Test framework: AGL-JTA
  - https://wiki.automotivelinux.org/agl-jta

Information

- Fuego
  - fuego-core:
    - https://bitbucket.org/fuegotest/fuego-core.git
    - e606654b8077 (core: update version numbers in common.sh)
  - fuego:
    - https://bitbucket.org/fuegotest/fuego.git
    - b5b69307f836 (install: fix debian jessie repositories)

- Target device in this slides
  - Raspberry Pi 3b
  - Rasbian, based on debian 9.4, Linux 4.14.34-v7+

Fuego testsuit

- Benchmark: 42
  - Dhrystone, GLMark, IOzone, Interbench, Java, OpenSSL, Stream, Whetstone, aim7, backfire, blobsallad, bonnie, cyclictest, dbench3, dbench4, dd, deadlinetest, ebizzy, ffsb, fio, fs_mark, gtkperf, hackbench, himeno, iperf, iperf3, linpack, lmbench2, migratetest, nbench_byte, netperf, netpipe, pmqtest, ptsematest, reboot, signaltest, sigwaittest, svsematest, sysbench, tiobench, vuls, x11perf
  - (exclude fuego selftests: 2)

Fuego testsuit

- Functional: 102 = 96 + 6
  - LTP, LTP_one_test, OpenSSL, acpid, aiostress, arch_timer, at, autopkgtest, bc, bgpd, bind, boost, brctl, bsdiff, busybox, bzip2, cmt, commonAPI_C++, commonAPI_Dbus, commonAPI_SomeIp, crashme, croco, cryptsetup, curl, dovecot, ethtool, expat, file, fixesproto, fontconfig, fsfuzz, ft2demos, fuse, giflib, glib, glib2, glibc, hciattach, hello_world, imagemagick, iperf3_server, ipmi, iptables, iputils, ipv6connect, jpeg, kernel_build, kmod, kselftest, libogg, libpcap, librsvg, libspeex, libtar, libwebsocket, libxml, linaro, linus_stress, lwip, mcelog, mesa_demos, module_init_tools, multipathd, neon, net-tools, netperf, nscd, nss, openct, openhpid, ospf6d, ospfd, pam, perl-xml-simple, pi_tests, pixman, pppd, protobuf, ptest, rmaptest, rpm, scifab, scrashme, sdhi_0, serial_rx, stress, synctest, tar, thrift, tiff, trousers, vconfig, vsomeip, xorg-macros, year2038, zlib
  - batch, batch_bc, batch_default, batch_hello, batch_nested, batch_smoketest
  - (exclude fuego selftests: 16)

LTP on Fuego

- has 14 specs
  - # ftc add-jobs -b yourboard -t Functional.LTP -s default
  - # ftc add-jobs -b yourboard -t Functional.LTP -s docker
  - # ftc add-jobs -b yourboard -t Functional.LTP -s selection
  - # ftc add-jobs -b yourboard -t Functional.LTP -s install
  - # ftc add-jobs -b yourboard -t Functional.LTP -s make_pkg
  - # ftc add-jobs -b yourboard -t Functional.LTP -s slectionwithrt
  - # ftc add-jobs -b yourboard -t Functional.LTP -s ltplite
  - # ftc add-jobs -b yourboard -t Functional.LTP -s ptsonly
  - # ftc add-jobs -b yourboard -t Functional.LTP -s smoketest
  - # ftc add-jobs -b yourboard -t Functional.LTP -s quickhit
  - # ftc add-jobs -b yourboard -t Functional.LTP -s rtonly
  - # ftc add-jobs -b yourboard -t Functional.LTP -s somefail
  - # ftc add-jobs -b yourboard -t Functional.LTP -s quickhitwithskips
  - # ftc add-jobs -b yourboard -t Functional.LTP -s security